CMPSC 465 Spring 2024 — Data Structures & Algorithms — Mehrdad Mahdavi and David Koslicki

Worksheet 4

Wednesday, February 7, 2024

1. **Recurrence Relations.** Give the big-Θ bound for each of the following recurrence relations. You may assume the Master Theorem gives a big-Θ bound (the proof is similar to the big-O proof shown in class) and that $T(1) = O(1)$. Justify each of your answers.

   a) $T(n) = 4T(n/2) + 42n$
   b) $T(n) = T(n-1) + n^3$
   c) $T(n) = 7T(n/2) + n^2 + \log n$
   d) $T(n) = T(n/3) + 5$

2. **Recurrence Relations.** What are the running times of each of these algorithms in big-O notation? Which is the fastest?

   a) Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.
   b) Algorithm B solves problems of size n by recursively solving two subproblems of size $n-1$ and then combining the solutions in constant time.
   c) Algorithm C solves problems of size n by dividing them into eight subproblems of size $\frac{n}{4}$, recursively solving each subproblem, and then combining the solutions in $O(n^3)$ time.

3. **Merge.** *A k-way merge operation.* Suppose you have k sorted arrays, each with n elements, and you want to combine them into a single sorted array of kn elements.

   a) Here's one strategy: Using the `merge` procedure, merge the first two arrays, then merge in the third, then merge in the fourth, and so on. What is the time complexity of this algorithm, in terms of k and n?
   b) Give a more efficient solution to this problem that runs in $O(nk \log k)$ using divide-and-conquer.

4. **Array Rotations.** Consider a rotation operation that takes an array and moves its last element to the beginning. After n rotations, an array $[a_0, a_1, a_2 \ldots a_{m-1}]$ of size m where $0 < n \leq m$, will become:

   $$[a_{m-n}, a_{m-n+1}, \ldots, a_{m-2}, a_{m-1}, a_0, a_1, \ldots a_{m-n-1}]$$

   Notice how a_{m-1} is adjacent to a_0 in the middle of the new array. For example, two rotations on the array $[1,2,3,4,5]$ will yield $[4,5,1,2,3]$.

   You are given a list of unique integers `nums`, which was previously sorted in ascending order, but has now been rotated an unknown number of times. Find the number of rotations in $O(\log n)$ time. *(Hint: consider Binary Search.)*

5. More Recurrence Relations. *Give the big-Θ bound for each of the following recurrence relations. You may assume that $T(1) = O(1)$. Justify each of your answers.*

a) $T(n) = 2T(2n/3) + T(n/3) + n^2$

b) $T(n) = 3T(n/4) + n\log n$